US007415531B2

(12) **United States Patent**
Musoll

(10) **Patent No.:** **US 7,415,531 B2**
(45) **Date of Patent:** **Aug. 19, 2008**

(54) **METHOD AND APPARATUS FOR PREDICTING CHARACTERISTICS OF INCOMING DATA PACKETS TO ENABLE SPECULATIVE PROCESSING TO REDUCE PROCESSOR LATENCY**

(75) Inventor: **Enrique Musoll**, San Jose, CA (US)

(73) Assignee: **Mips Technologies, Inc.**, Mountain View, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 588 days.

(21) Appl. No.: **09/935,446**

(22) Filed: **Aug. 22, 2001**

(65) **Prior Publication Data**

US 2003/0041168 A1 Feb. 27, 2003

(51) **Int. Cl.**
G06F 15/173 (2006.01)

(52) **U.S. Cl.** **709/238**; 709/239; 709/240

(58) **Field of Classification Search** 709/238; 370/94.1; 364/900
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,056,058 A * 10/1991 Hirata et al. 709/230
5,260,942 A * 11/1993 Auerbach et al. 370/474
5,557,608 A * 9/1996 Calvignac et al. 370/389
5,857,201 A * 1/1999 Wright et al. 707/104.1
6,243,610 B1 * 6/2001 Iino et al. 700/1
6,359,883 B1 * 3/2002 Lechleider 370/389
6,430,156 B1 * 8/2002 Park et al. 370/232
6,622,173 B1 * 9/2003 Luke 709/237

FOREIGN PATENT DOCUMENTS

WO WO0111834 A1 * 2/2001

OTHER PUBLICATIONS

Krishnamurthy, Balachander and Rexford, Jennifer, "En Passant: Predicting HTTP/1.1 Traffic," 1999, IEEE, vol. 3, pp. 1768-1773.*
Kim, Tae-eun et al., "Improving Congestion Control Performance Through Loss Differentiation," Oct. 1999, IEEE, pp. 412-418.*
Matic, Victor et al. "Predictive Playout Delay Adaptation for Voice over Internet," 2000, IEEE, pp. 348-351.*
Yuan, Chin and Silvester, John A. "Queueing Analysis of Delay Constrained Voice Traffic in a Packet Switching System," 1989, IEEE, vol. 7, No. 5, pp. 729-738.*
Enrique Musoll Prediction of Packet Flow, Packet Header and Packet Payload Jun. 14, 2001, Disclosure Document #495342, USPTO.

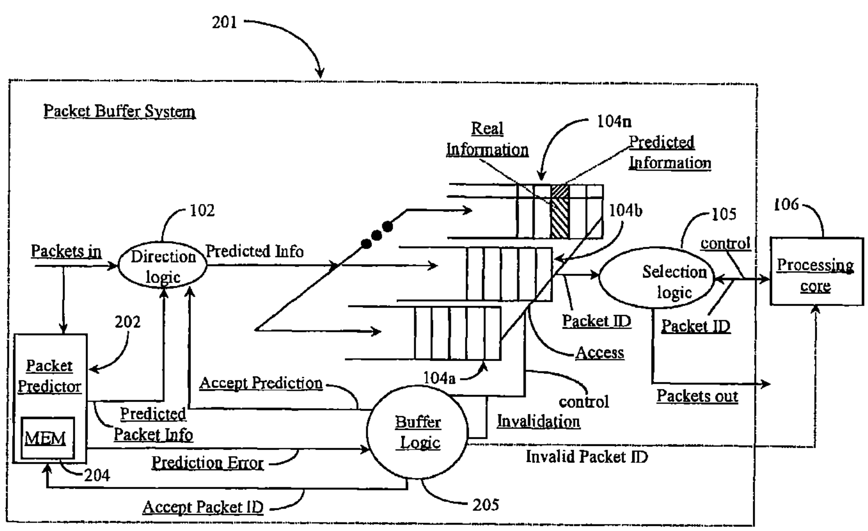
* cited by examiner

Primary Examiner—Jason Cardone
Assistant Examiner—Azizul Choudhury
(74) *Attorney, Agent, or Firm*—Sterne Kessler Goldstein Fox PLLC

(57) **ABSTRACT**

A system for processing data packets in a data packet network has at least one input port for receiving data packets, at least one output port for sending out data packets, a processor for processing packet data, and a packet predictor for predicting a future packet based on a received packet, such that at least some processing for the predicted packet may be accomplished before the predicted packet actually arrives at the system. The system is used in preferred embodiments in Internet routers.

22 Claims, 5 Drawing Sheets



DESCRIPTION OF THE PREFERRED EMBODIMENTS

As described in the background section of this specification, current art packet processors, operating typically in data routers, operate only on packets received to be routed. The inventor, according to an embodiment of the present invention, provides an apparatus and method that enables a processor to predict characteristics of a next incoming data packet and initiate speculative processing before the actual next packet arrives for processing. The method and apparatus of the present invention is described in enabling detail below.

FIG. **1** is block-diagram illustrating components of a typical network processing system according to prior-art. A packet buffer or queuing system **101** is illustrated in this example of prior art as a typical system present in data packet processors. Packet buffering system **101** comprises a direction logic block **102** adapted for receiving data packets at ingress and determining which of a plurality of data queues will be selected for holding the packet data prior to processing. In this example, there are a plurality of logically-illustrated queues **104***a*, **104***b* to **104***n*. It will be appreciated by one with skill in the art of data processors that there typically may be more than three queues associated with a data packet processor used for data routing on a data network.

In this example, an incoming data packet labeled Packet Info has been entered in queue **104***n* as indicated by cross-hatching. The data actually stored in such a queue can include simply packet identifiers and a packet header to be processed, or a complete and full data packet. This option is largely design dependant. The set of queues **104***a-n* has an associated buffer control logic **103** that is connected by control path to a common access path or line shared by each queue. The method of control and communication between buffer logic **103** and queue set **104***a-n* is only logically represented in this example. Actual physical links and structures may vary according to design. It will be appreciated by the skilled artisan that multiple ingress ports may share one set of data queues **104***a*-**104***n* and that it is assumed that data packets are serially enqueued and dequeued.

Selection logic **105** is illustrated within packet buffer system **101** and is adapted to manage how processed packets are selected to be sent out of queue to egress of the processing system after processing is complete. Selection logic **105** is illustrated as connected by control path to a common access line shared by all of the queues in set **104***a-n* as was described above with reference to direction logic **103**. A processing core **106** is illustrated in this embodiment and is adapted to process data packet information while data packets are in the system. Processing core **106** is logically connected through selection logic **105** to queue set **104***a-n* by a control line.

In typical prior-art processing, packets arrive through ingress of the system as illustrated herein by the label Packets In, and are buffered in any one of queues comprising set **104***a-n* according to direction logic **102**. At least a packet identifier including a queue address location identifier labeled herein simply Packet ID is made available to processing core **106**. Core **106** processes the packet information according to applicable software. It is noted in this prior-art example that processing by core **106** cannot begin until an actual data packets or sufficient information of one is enqueued in one of queues **104***a-n* and is identified and registered within the system.

It is known that data traffic over a data-packet-network such as the Internet network typically arrives at processors in a series of data bursts. This means that the processing workload over time of core **106** will experience peaks, valleys, and perhaps periods of idleness. These periods of low workload and idle times are unavoidable in the current art. The goal of the present invention is to utilize low workload and idle times for speculative data processing on future data packets yet to arrive.

FIG. **2** is a block-diagram illustrating components of a novel network processing system according to an embodiment of the present invention. In this example, a packet buffering system **201** is provided with a capability of predicting characteristics of some data packets before they arrive to the processing system. System **201** comprises queues **104***a-n* as described with reference to FIG. **1** above. Direction logic **102** and selection logic **105** are also present in this example as is processing core **106**, and these elements are analogous to the components with the same element numbers previously described.

A novel hardware mechanism labeled herein Packet Predictor **202** is provided within system **201** and enables prediction of data packet characteristics before some packets actually arrive through ingress of the processing system. Data packet predictor **202** is a front-end hardware implementation that generates speculative packet information for a virtual data packet (predicted data packet). A packet prediction may be triggered by any one of several events and conditions, such as detection of idle processor time. A good time, however, and a trigger used in a preferred embodiment is when a real data packet is received by the processing system.

Packet predictor **202** has, in this example, a dedicated memory (MEM) **204** provided therein and adapted to store historical data regarding real data packets previously processed within the system and historical data about successful instances of predicted data packets within the system, wherein the speculative processing results associated with a predicted packet, backed up by a real packet, were correct enough to send the real packet out of the system requiring little or no processing of the information associated with the real packet. MEM **204** can be a flash type MEM, ROM, RAM, or any other type of usable memory sufficient in size to hold at least a historical record covering a pre-defined number of data packets.

In a preferred embodiment, MEM **204** stores a revolving history record that is updated periodically, whether or not the processing was "real" or "virtual". For example, MEM **204** may store historical data covering the last 10 data packets received, and also the practical result of the last ten data packets predicted. In other embodiments the history record could cover many more, or fewer data packets, both real and virtual. In an alternative embodiment of the invention, MEM **204** may be implemented externally from predictor **202** or from buffer system **201**, or even externally from the router without departing from the spirit and scope of the present invention. For example, MEM **204** may be an assigned portion of existing memory within the processing system such as queue memory or processing core memory. There are many possibilities.

A buffer logic **205** is provided within packet buffering system **201** and adapted to control queue-state reporting and management of queues **104***a-n* similarly to buffer logic **103** described above with reference to FIG. **1**. In this example logic **205** is enhanced to manage queue set **104***a-n* according to additional predictive capabilities of the present invention. Logically speaking, buffer logic **205** has control connection to a shared access line of queue set **104***a-n* as well as a control and communication connection to packet predictor **202**. Additionally, there are illustrated connections between buffer logic **205** and direction logic **102** and between buffer logic